**Hybrid 수송모델을 기본으로한 간선 차량 배치 최적화 모델링**

한진물류연구원 인턴 이성우

**1. 연구의 개요**

한진, 대한통운 등의 택배사들은 Hub & Spoke 수송 모델과 Point to Point 수송 모델의 장점을 더한 절충형 모델인 Hybrid 수송 모델을 사용하고 있다. 따라서, 간선노선의 경우의 수가 많아지고, 모델의 복잡도는 올라가며 의사결정 변수가 증가하여 비용과 인력의 최적화에 어려움을 겪고 있다. 이를 해결하고자 ‘배차 최적화’에 초점을 둔 수리 모형을 구성하고, 이를 python과 google ortools의 algorithm을 이용하여 최적값을 찾아보았다.

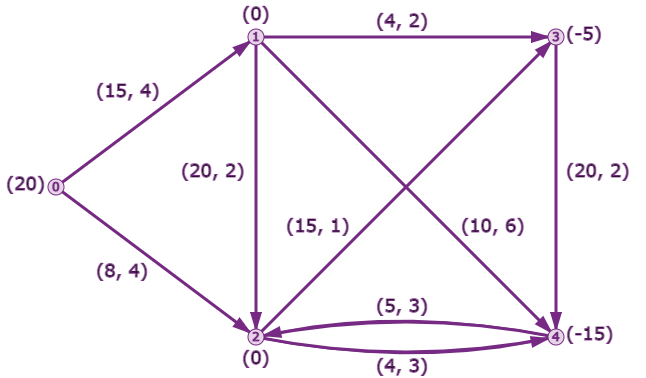
우선 선행연구논문을 찾아보며 관련 지식을 습득하였는데, 참고한 논문 중, ‘택배 서비스 네트워크 설계를 위한 수리적 모형 개발 (김성우 외 7인 저)’이 많은 도움이 되었다. 해당 논문은 배차에 관한 내용을 다루지는 않았지만, 모델의 변수 선정에 많은 인사이트를 얻을 수 있었고, 수리 모형의 아이디어를 얻을 수 있었다.

6주안에 과업을 해결하기 위해 코딩과 알고리즘을 단순화하여 진행하였다. 분석은 python을 이용하였고, solver는 google 사에서 제공하는 Minimum Cost Flow와 MIP problem Solver를 이용하였다. 최초의 구성은, 송장번호를 기준으로 Unique한 각각의 택배와 차량을 전부 variable로 두고 경유 수송 모델 코딩을 진행하려 하였으나, 이는 모델의 구성 시간도 오래 걸리고, 지나치게 복잡한 알고리즘은 많은 컴퓨팅 파워를 요구하여 빠른 시간 내에 최적화된 답을 원하는 현장에서는 적합하지 않을 수 있다고 생각하여, 효율적인 모델 구성에 초점을 두고 현재의 모델을 구성하기로 하였다.

**2. 모델링**

모델 구성은 다음과 같이 진행하였다. 우선 각각의 Link(모델에선 Arc로 표현)에 단위 cost와 각 터미널의 Capacities를 고려하여 최적화된 물동량을 배정한다.(이때 Minimum Cost Flow Algorithm이 사용되며, 단위 cost와 Capacities는 별도의 Algorithm이 필요) 이렇게 배정된 Link별 물동량을 새로운 DataFrame으로 구성하여 해당 Data를 가지고 IP모델을 구성, MIP problem Solver로 최적해를 찾았다. 마지막으로 각 Link별로 배차된 차량이 왕복 수송이 가능 여부를 확인함으로 모델이 완성된다.

Minimum Cost Flow Algorithm은 다음과 같다. start\_nodes, end\_nodes, capacities, unit\_costs를 결정하고 각 Node에 supplies(node out은 양의 정수, node in은 음의 정수로 할당)를 할당하면 ortools.graph의 pywrapgraph패키지 안의 pywrapgraph.SimpleMinCostFlow() 함수를 통해 Minimum cost와 Optimal Cost를 계산해 준다.



min cost flow의 예시 (Google OR-Tools)

주어진 data는 총 6개의 터미널로 구성되어 있고, 2번과 4번 터미널은 Hub, 나머지 터미널은 분류 기능이 없는 Sub로 가정한다. 모델링에 앞서 세 가지 전제를 하였는데, 첫째, 모든 택배는 동일한 중량과 동일한 사이즈를 가진다. 둘째, 택배차량은 11ton, 8ton, 2.5t만이 존재한다. 셋째, 1차 물동량 배치는 단위 요금으로 정하고, 차량 배차 시 각 링크에 걸리는 코스트 최소화하여 최적화를 진행한다. 여기서 단위 요금은 data가 없어 임의의 값으로 지정하였고, Hub에 걸리는 Link의 경우 저렴하고 더 많은 Capa를 가지도록 설정하였다.

첫번째 Algorithm인 Minimum Cost Flow를 실행하면 다음과 같은 data를 얻을 수 있다.

****

node와 node 사이의 link에 걸리는 물동량 (박스단위)

해당 DataFrame을 바탕으로 MIP Solver를 이용하여 11톤, 8톤, 2.5톤의 차량을 각각의 링크에 배차한다.

MIP problem은 다음과 같이 진행하였다. ortools.linear\_solver의 pywraplp패키지 안의 SCIP Solver를 이용하였으며, Vehicle을 Decision Variable Binary(배차가 되면 1, 아니면 0)로 설정하였다. Vehicle의 notation은 i(link 번호), t(type, 11ton, 8ton, 2.5ton으로 구분), idx(차량 index)로 부여하였고 3-dimensional으로 모형의 복잡도를 낮추었다.

이를 바탕으로 두번째 Algorithm인 MIP Solver를 실행하면 다음과 같은 data를 얻을 수 있다.



각 Link에 Cost를 최소로 하는 Vehicle 할당

위와 같이 각각의 링크에 어떤 타입의 몇 대의 차량을 배차해야 Total Cost를 최소로 하는지 Solution을 구해준다. 마지막 열의 loss ratio는 배차된 차량이 총 적재 용량 대비 적재된 용량을 1에서 뺀 것으로 전체 적재량 중 적재되지 않은 양을 나타낸다.

마지막으로 위의 data를 바탕으로 왕복 차량에 대한 분석을 수행하였다. 예를 들어 0번 노드에서 2번노드로 가는 차량과 2번 노드에서 0번 노드로 가는 차량이 겹친다면, 각각 편도 운행을 하는 것이 아닌, 왕복 운행을 시행할 수 있다. (여기선 외부 Algorithm이 아닌 직접 코딩한 Algorithm을 사용) 분석 결과는 다음과 같다.



왕복 수송 노선



편도 수송 노선



왕복 수송 노선에서의 왕복편 배차량

위의 data를 바탕으로 왕복 노선의 경우 정규편을 배차하는 등의 의사결정에 도움을 줄 수 있다.

**3. 연구의 한계점**

해당 연구는 많은 한계점을 가지고 있다. 우선 실제 데이터를 가지고 연구를 진행하지 못했다. 실제 현장 데이터를 얻는 과정에서 문제가 생겨 이번 연구는 가상의 데이터를 가지고 진행하였다. 따라서 물동량이나 비용 등이 임의로 지정된 것이 많아, 결과물의 신뢰성이 아주 높지 못하다. 이러한 가정을 최대한 줄이기 위해 터미널 조업료를 생략한 것도 한계점으로 남는다.

다음으로 수송 시간과 터미널 조업 시간을 고려하지 못하였다. 최초의 연구과정에선 시간 변수를 고려하려 연구를 진행하였으나 이는 첫째로, 모형의 복잡도가 증가하여 6주 내에 의미 있는 결과물을 도출하기 어렵다는 결론을 내렸다. 둘째로, 위에서 서술한 바와 같이, 빠른 의사결정을 원하는 현장에서 많은 컴퓨팅 파워를 소요하여 긴 시간을 필요로 하는 모델은 현실적으로 적합하지 못하다는 결론을 내려 해당 모델로 연구를 진행하게 되었다. 하지만 연구자의 입장에서 시간 변수를 고려하는 것이 조금 더 현실의 모형을 반영하는 방향이고 더 의미 있는 분석이 가능하다는 점에서 추후 연구에서 이를 고려하였으면 한다.